

This is a section of [doi:10.7551/mitpress/9153.001.0001](https://doi.org/10.7551/mitpress/9153.001.0001)

# **Certified Programming with Dependent Types**

## **A Pragmatic Introduction to the Coq Proof Assistant**

**By: Adam Chlipala**

### **Citation:**

*Certified Programming with Dependent Types: A Pragmatic Introduction to the Coq Proof Assistant*

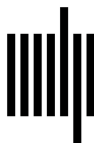
**By: Adam Chlipala**

**DOI: 10.7551/mitpress/9153.001.0001**

**ISBN (electronic): 9780262317870**

**Publisher: The MIT Press**

**Published: 2022**



**The MIT Press**

# Index

- `.dir-locals.el` file, 11
- `.emacs` file, 11
  
- abstract type, 380
- accessibility relation, 135
- ACL2, 1, 3–5
- adaptive proof scripts, 371
- Agda, 6
- algebraic datatypes, 14, 43
- array bounds checks, 155
- `as` clause, 160
- axiom K, 217, 280
- axiom of choice, 270
- axioms, 216, 265
  
- backtracking, 289
- beta reduction, 208
- bisimulation, 103
- bullets, 367
  
- Calculus of Constructions, 15
- Calculus of Inductive  
  Constructions, 15
- certified program, 2
- certifying program, 2
- CIC, *see* Calculus of Inductive  
  Constructions
- class (in set theory), 252
- classical logic, 80
  
- co-induction principles, 102
- co-inductive big-step operational  
  semantics, 107
- co-inductive predicates, 99
- co-inductive types, 94, 206
- co-recursive definitions, 94
- CoC, *see* Calculus of  
  Constructions
- computability, *see* decidability
- conclusion, 20
- constructive logic, 45, 80, 266
- context patterns, 311
- continuation-passing style, 322
- convoy pattern, 172, 276
- `coq_makefile`, 384
- `coqdoc`, 14
- CoqIDE, 9
- Curry-Howard correspondence,  
  42, 73, 93, 263
- currying, 151
  
- datatype-generic programming,  
  233
- de Bruijn criterion, 4
- decidability, 80
- declarative proof scripts, 370
- deep embedding, 133, 280
- definitional equality, 207
- delta reduction, 208
- dependent pair, 117

- dependent pattern matching, 30, 61, 159
- dependent types, 113
- deriving clauses, 233
- discriminee, 159
- domain theory, 140
  
- elimination, 262
- Epigram, 6
- existential quantification, 81
- extensionality, 139
- extensionality of function
  - equality, 229
- extraction, *see* program
  - extraction
  
- failure monad, 126
- first-order syntax, 393
- free variable, 20
- fully automated proofs, 363
- functional extensionality, 231, 389
- functor, 380
  
- GADTs, *see* generalized algebraic datatypes
- Gallina, 16, 251
- Gallina operators
  - ++, 18
  - /\, 57
  - ::, 17
- Gallina terms
  - Acc, 135
  - and, 57, 65
  - app, 51
  - as, 61
  - bool, 46
  - cofix, 211
  - cons, 51
  - Empty\_set, 45
  - eq, 83
  - eq\_ind\_r, 225
  - eq\_rect\_eq, 216
  - eq\_refl, 83
  - ex, 81
  - exists, 81
  - False, 42
  - false, 46
  - fin, 185
  - Fix, 136
  - fix, 61, 209
  - for, 63
  - forall, 81
  - hlist, 189, 235
  - I, 42
  - if, 46
  - ilist, 185, 235
  - in, 156
  - index, 348
  - internal\_JMeq\_rew\_r, 225
  - JMeq, 222
  - length, 51
  - list, 51
  - maybe, 125
  - member, 189
  - nat, 47
  - negb, 46
  - nil, 51
  - 0, 47
  - option, 17
  - or, 77
  - plus, 48
  - pred, 47
  - prod, 57, 76
  - proj1\_sig, 117
  - Prop, 44
  - return, 61
  - S, 47
  - Set, 14, 44
  - sig, 116
  - sigT, 171
  - sum, 77
  - sumbool, 121

- sumor, 125
- True, 42
- true, 46
- tt, 43
- UIP\_refl, 216
- unit, 43
- varmap, 349
- with, 63
- generalized algebraic datatypes, 29, 43
- generic programming, 233
- GHC Haskell, 29
- Girard's paradox, 252
- graphical interfaces to Coq, 9
- guardedness condition, 95
  
- Haskell, 29, 43, 58, 93, 321
- head symbol, 304
- head-normal form, 184
- heterogeneous equality, 222
- higher-order abstract syntax, 59, 397
- higher-order syntax, 397
- higher-order unification, 61, 115
- higher-order vs. first-order languages, 3
- Hilbert's epsilon operator, 272
- hint databases, 294
- HOAS, *see* higher-order abstract syntax
- HOL, 4
- hypotheses, 20, 23
  
- implicit arguments, 66, 259
- impredicative Set, 272
- impredicativity, 255, 263
- in clause, 160
- inconsistent axioms, 266
- index function, 196
- induction principles, 44, 60
- inference rules, 83
- intensional type theory, 218
- interactive proof-editing mode, 19
- interpretation function, 343
- interpreters, 14, 191, 200
- intro pattern, 108, 365
- intuitionistic logic, *see* constructive logic
- iota reduction, 208
- Isabelle/HOL, 2, 5
- Isar, 370
  
- John Major equality, 222
- judgment, 83
  
- lambda calculus, 59, 191
- large inductive types, 179, 255
- law of the excluded middle, 80, 265
- laziness, 93
- length-indexed lists, 155
- linear arithmetic, 309
- logic programming, 287
- Ltac, 5, 16
  
- Makefile, 384
- metalanguage, 29, 397
- ML, 43, 58
- module systems, 233
- monad, 126, 142, 321
  
- natural deduction, 83, 312
- nested inductive type, 64, 197
- notation scope delimiter, 162
- notation scopes, 32
- Nqthm, 1
  
- Obj.magic, 177
- object language, 29, 397
- OCaml, 113

- opaque, 119
- opaque ascription, 381
  
- parameters, 161
- parametric higher-order abstract
  - syntax, 397
- parametric polymorphism, 233
- parametricity, 408
- Park's principle, 103
- phase distinction, 156
- PHOAS, *see* parametric
  - higher-order abstract
    - syntax
- polymorphism, 51
- positivity requirement, 59
- predicativity, 253
- Presburger arithmetic, 309
- primitive recursion, 133
- principal types, 15
- productivity, 98
- Program, 121
- program extraction, 37, 80, 113, 262
- Prolog, 83, 287
- proof by reflection, 6, 339
- Proof General, 9, 11–12
- proof irrelevance, 74, 268
- proof term, 43
- proof-carrying code, 275
- propositional equality, 211
- PVS, 4, 5
  
- recursion principles, 61
- recursion schemes, 236
- recursive type definition, 193
- recursively nonuniform
  - parameters, 149
- reduction strategy, 16
- reflection, 6
- reflexive inductive type, 198
- reification, 343
  
- relative consistency, 15
- return** clause, 160
- rule induction, 88
- Russell's paradox, 255
  
- sections, 52
- set theory, 251
- setoids, 310
- shallow embedding, 133
- sigma type, 117
- stratified type systems, 157
- strengthening the induction
  - hypothesis, 19
- strict positivity requirement, 59, 397
- strong normalization, 15
- structural induction, 20
- subgoals, 19
- subset types, 116
  
- tactical, 310
- tactics, 20
  - abstract**, 120, 380
  - apply**, 55, 91
  - assumption**, 77
  - auto**, 86, 289
  - autorewrite**, 304
  - cbv**, 207
  - change**, 70, 230, 347
  - congruence**, 49, 303, 309
  - constr**, 318
  - constructor**, 75
  - crush**, 10, 26
  - crush'**, 131
  - debug**, 293, 379
  - decide equality**, 123
  - dep\_destruct**, 166
  - dependent destruction**, 166, 277
  - destruct**, 44, 108
  - discriminate**, 47, 69

- do, 333
- eapply, 290
- eauto, 91, 291
- elimtype, 75
- eval, 331
- evan, 331
- exact, 276, 312, 341
- exists, 81
- fail, 314
- field, 348
- first, 314
- firstorder, 82
- fold, 23, 25
- fourier, 309
- fresh, 332
- generalize, 169, 220, 314
- hnf, 184
- idtac, 313, 314
- induction, 20, 26, 44, 365
- info, 289, 375
- info\_auto, 289
- injection, 49, 69
- instantiate, 336
- intro, 70
- intros, 20, 23, 26, 277
- intuition, 78, 309
- inversion, 85
- lazy, 208
- left, 78
- match, 91, 309
- omega, 300, 309
- pattern, 226, 245
- pose, 319
- progress, 327
- quote, 348
- red, 69
- refine, 118, 138
- reflexivity, 23, 25, 27
- repeat, 310
- rewrite, 24, 25, 27, 70
- right, 78
- ring, 301, 309, 348
- semicolon, 26
- simpl, 22, 25, 48
- simple apply, 329
- simple destruct, 214
- solve, 335
- specialize, 312
- split, 77
- tauto, 78
- trivial, 49
- type of, 314
- unfold, 21, 23, 25
- using, 68
- with, 90
- tagless interpreters, 161
- termination checking, 16, 59
- theory of equality and
  - uninterpreted functions, 49
- thunks, 380
- transparent, 119
- transparent ascription, 381
- trusted code base, 275
- Twelf, 4, 5, 59
- type classes, 233
- type hierarchy, 251
- type inference, 15, 61
- unicity of identity proofs, 269
- unification, 289
- unification variable, 290
- universe inconsistency, 254
- universe polymorphism, 258
- universe types, 233
- universe variable, 253
- variable binding, 389
- Vernacular commands, 16
  - Abort, 25
  - Axiom, 265
  - Check, 24
  - CoFixpoint, 94
  - CoInductive, 94

- Debug On, 374
- Declare Module, 381
- Defined, 119, 137
- Definition, 14
- Eval, 16
- Example, 57
- Extract Inductive, 123
- Extraction, 37, 113
- Fixpoint, 16
- Function, 140
- Guarded, 100
- Hint Constructors, 289, 302
- Hint Extern, 69, 298
- Hint Immediate, 291, 302
- Hint Resolve, 292, 302
- Hint Rewrite, 36, 304
- Hint Unfold, 302
- Hypothesis, 62
- Implicit Arguments, 52
- Import, 385
- Inductive, 14
- Lemma, 19
- Load, 385
- Locate, 65
- Ltac, 175
- Module, 381
- Module Type, 381
- Obligation Tactic, 121
- Open Scope, 178
- Parameter, 266
- Print Assumptions, 224, 267
- Print Universes, 256
- Program Definition, 121
- Program Fixpoint, 140
- Qed, 26, 380
- Recursive Extraction, 177
- Require, 385
- Require Export, 386
- Require Import, 380
- Restart, 46
- Scheme, 55
- SearchRewrite, 24, 291
- Section, 52
- Set Implicit Arguments, 53
- Set Printing All, 260
- Set Printing Universes, 252
- Show Proof, 296
- Theorem, 19
- Time, 293, 377
- Variable, 52
- well-founded recursion, 134
- well-founded relation, 134
- Zermelo-Fraenkel set theory, 15
- zeta reduction, 208

© 2013 Massachusetts Institute of Technology

All rights reserved. No part of this book may be reproduced in any form by any electronic or mechanical means (including photocopying, recording, or information storage and retrieval) without permission in writing from the publisher.

MIT Press books may be purchased at special quantity discounts for business or sales promotional use. For information, please email [special\\_sales@mitpress.mit.edu](mailto:special_sales@mitpress.mit.edu) or write to Special Sales Department, The MIT Press, 55 Hayward Street, Cambridge, MA 02142.

This book was set in 10/13 Lucida Bright by the author using L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>. Printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Chlipala, Adam, 1981–

Certified programming with dependent types : a pragmatic introduction to the Coq proof assistant / Adam Chlipala.

p. cm

Includes bibliographical references and index.

ISBN 978-0-262-02665-9 (hardcover : alk. paper)

1. Automatic theorem proving—Computer programs. 2. Computer programming. 3. Coq (Electronic resource) I. Title.

QA76.9.A96C45 2013

005.1—dc23

2013012837

10 9 8 7 6 5 4 3 2 1