

---

## learn-tt

Lots of people seem curious about type theory but it's not at all clear how to go from no math background to understanding "Homotopical Patch Theory" or whatever the latest cool paper is. In this repository I've gathered links to some of the resources I've personally found helpful.

### A Disclaimer

At this point `learn-tt` is a few years old and I can write with slightly more confidence than when I first created this document. I still stand by the fact that the links in this list have helped me learn ideas that would have been difficult to find elsewhere. At the same time, I worry that this list carries more of a ring of authority than I wish it did, particularly now that it is relatively popular. I learn more about type theory every day (admittedly, some days more slowly than I wish) and my views on what constitutes a good explanation, a good approach, or even a good type theory have changed in small and large ways since I first compiled these resources.

I toyed with the idea of deleting this repository because I have worried whether or not presenting my own learning path does more harm than good. I have decided to leave it around but to add a disclaimer instead.

What follows below should not be construed as some *blessed path* for learning type theory. You may find it better to skim this list or simply snort and ignore it entirely. My process for learning continues to be distinctly wandering and non-linear. Someone with different goals than me would find some of these links useless and I would not be nearly so bold as to claim that these resources are canonical, necessary, or even helpful for everyone. I can only hope that you enjoy reading them as much as I have.

Danny

### The Resources

#### Textbooks

- Practical Foundations of Programming Languages (PFPL)

I reference this more than any other book. It's a very wide ranging survey of programming languages that assumes very little background knowledge. A lot people prefer the next book I mention but I think PFPL does a better job explaining the foundations it works from and then covers more topics I find interesting.

- 
- Online copy (2nd Edition Preview)
  - Dead-tree copy (2nd Edition)

- Types and Programming Languages (TAPL)

Another very widely used introductory book (the one I learned with). It's good to read in conjunction with PFPL as they emphasize things differently. Notably, this includes descriptions of type inference which PFPL lacks and TAPL lacks most of PFPL's descriptions of concurrency/interesting imperative languages. Like PFPL this is very accessible and well written.

- Online supplements
- Dead-tree copy

- Advanced Topics in Types and Programming Languages (ATTAPL)

Don't feel the urge to read this all at once. It's a bunch of fully independent but excellent chapters on a bunch of different topics. Read what looks interesting, save what doesn't. It's good to have in case you ever need to learn more about one of the subjects in a pinch.

- Online supplements
- Dead-tree copy

## **Proof Assistants**

One of the fun parts of taking in an interest in type theory is that you get all sorts of fun new programming languages to play with. Some major proof assistants are

- Coq

Coq is one of the more widely used proof assistants and has the best introductory material by far in my opinion.

- Official site
- Software Foundations
- Certified Programming with Dependent Types
- The paper on the calculus of constructions
- A paper on the calculus of inductive constructions (What Coq is based on)

- Lean

Lean is one of the newer proof assistants on the scene. To be perfectly honest I haven't done a lot of proving in Lean yet but it seems neat. (If you have any resources you'd like to add to this list, please let me know)

- 
- Official site
  - Logic and Proof Textbook

- Agda

Agda is in many respects similar to Coq, but is a smaller language overall. It's relatively easy to learn Agda after Coq so I recommend doing that. Agda has some really interesting advanced constructs like induction-recursion.

- Official site
- Tutorial
- Records tutorial
- Conor McBride's fun Agda code
- Martín Hötzel Escardó's notes on Univalent Mathematics in Agda

- Idris

It might not be fair to put Idris in a list of "proof assistants" since it really wants to be a proper programming language. It's one of the first serious attempts at writing a programming language with dependent types *for actual programming* though.

- Official site
- An actual introductory book
- Quick tutorial
- A list of talks on Idris
- David Christiansen's cool talk

- Twelf

Twelf is by far the simplest system in this list, it's the absolute minimum a language can have and still be dependently typed. All of this makes it easy to pick up, but there are very few users and not a lot of introductory material which makes it a bit harder to get started with. It does scale up to serious use though.

- Official site
- Wiki Tutorials
- My tutorial
- The paper on LF, the underlying system of Twelf

## Type Theory

- The Works of Per Martin-Löf

---

Per Martin-Löf has contributed a *ton* to the current state of dependent type theory. So much so that it's impossible to escape his influence. His papers on Martin-Löf Type Theory (he called it Intuitionistic Type Theory) are seminal.

If you're confused by the papers above read the book in the next entry and try again. The book doesn't give you as good a feel for the various flavors of MLTT (which spun off into different areas of research) but is easier to follow.

- 1972
- 1979
- 1984
- The Complete Works of Per Martin-Löf

- Programming In Martin-Löf's Type Theory

It's good to read the original papers and here things from the horses mouth, but Martin-Löf is much smarter than us and it's nice to read other people explanations of his material. A group of people at Chalmers have elaborated it into a book.

- Online link

- The Works of John Reynolds

John Reynolds' works are similarly impressive and always a pleasure to read.

- Types, Abstraction and Parametric Polymorphism (Parametricity for System F)
- A Logic For Shared Mutable State
- Course notes on separation logic
- Course notes on denotational semantics

- Homotopy Type Theory

A new exciting branch of type theory. This exploits the connection between homotopy theory and type theory by treating types as spaces. It's the subject of a lot of active research but has some really nice introductory resources even now.

- The HoTT book
- Student's Notes on HoTT
- Materials for the Schools and Workshops on UniMath

## Proof Theory

- Frank Pfenning's Lecture Notes

---

Over the years, Frank Pfenning has accumulated lecture notes that are nothing short of heroic. They're wonderful to read and almost as good as being in one of his lectures.

- Introductory Course
- Linear Logic
- Modal Logic

- Jean-Yves Girard's Books

Girard, one of the most influential logicians of our time, has written several excellent texts on proof theory and logic. My ability to appreciate them is somewhat hampered by a language barrier but what work is available in English I have enjoyed.

- Proofs and Types
- The Blind Spot: Lectures on Logic
- Mustard Watches: An Integrated Approach to Time and Food

## Category Theory

Learning category theory is necessary to understand some parts of type theory. If you decide to study categorical semantics, realizability, or domain theory eventually you'll have to buckledown and learn a little at least. It's actually really cool math so no harm done!

- Category Theory in Context

A newly released textbook on category theory with a focus on using representable functors as a tool to place various concepts of category theory in a coherent framework. This has the substantial advantage of being freely available online! It's also published by Dover so the actual book itself is remarkably cheap.

- Online version
- Dead-tree version
- The author's post on the book

- Practical Foundations of Mathematics

This book does an excellent job of tying together general mathematics into the framework of category theory. It accordingly covers a large basis of *math* outside of the field of category theory. It contains a large amount of categorical logic which warrants its inclusion in this list and is one of the more approachable texts on categorical logic. At least for me.

- HTML version
- Dead-tree version

---

- Category Theory

One of the better introductory books to category theory in my opinion. It's notable in assuming relatively little mathematical background and for covering quite a lot of ground in a readable way.

- Dead-tree version

- Ed Morehouse's Category Theory Lecture Notes

Another valuable piece of reading are these lecture notes. They cover a lot of the same areas as "Category Theory" so they can help to reinforce what you learned there as well giving you some of the author's perspective on how to think about these things.

- Online copy

- Categorical Logic and Type Theory

This book is honestly quite difficult to get through, but it's an absolutely indispensable resource for folks interested in categorical logic. More generally, this book contains one of the few coherent and comprehensive accounts of how to model type theory categorically. It is *not* a book to learn category theory or type theory from, it demands a good understanding of both since it's focused on applying category theory, not explaining it so much. This is also the book to read if you're interested in understanding the theory of fibered categories in general (the style of categorical semantics that it uses).

- Jacob's thesis, containing much of what went into the book
- A definitely not suspicious online copy
- Dead-tree copy

- Introduction to Higher-Order Categorical Logic

This is a relatively short book on categorical logic that introduces all the basic concepts you needed to model simple higher-order logics in category theory. It is *much* easier reading than Categorical Logic and Type Theory but correspondingly less comprehensive. It focuses mainly on modeling the simply typed lambda calculus in cartesian closed categories and then on modeling a richer type theory internally to a topos. It provides a basic explanation of topos theory so it's intelligible having read an introductory category theory book.

- Dead-tree copy

- Sheaves in Geometry and Logic

This is not an ideal first book on category theory by any stretch. It merits inclusion because there are deep and interesting relationships between topos theory and type theory and this is

---

one of the more approachable introductions. Some knowledge of topology would be helpful in understanding some of the examples in this books but I am told it is possible to muscle your way through without it.

- Dead-tree version

## Other Goodness

- Gunter’s “Semantics of Programming Language”

While I’m not as big a fan of some of the earlier chapters, the math presented in this book is absolutely top-notch and gives a good understanding of how some cool fields (like domain theory) work.

- Dead-tree version

- Abramsky and Jung’s “Domain Theory”

This what I reference nowadays for domain theory. It’s a very good (if a little dense) introduction covering all the basic mathematics necessary to work with domains productively. It should definitely be possible to follow if you’ve read some of Gunter’s book.

- CiteSeerX link

- Realizability: An Introduction to Its Categorical Side

Categorical realizability is a fascinating area of overlap between type theory and category theory that, frustratingly, lacks many approachable introductions. van Oosten’s book does a good job going through the basic aspects of categorical realizability. It is heavily dependent on knowledge of category theory though, I would recommend making it through Sheaves and Geometry and Logic (see above) or something equivalent first.

- Dead-tree version

- OPLSS

The Oregon Programming Languages Summer School is a 2 week long bootcamp on PLs held annually at the university of Oregon. It’s a wonderful event to attend but if you can’t make it they record all their lectures anyways! They’re taught by a variety of lecturers but they’re all world class researchers.

- 2012
- 2013
- 2014

- 
- 2015
  - 2016
  - 2017
  - 2018