I'm stuck in the learning process, general help?

help

Niro March 3, 2023, 4:12pm 1

Hey everyone, yet another new learner here. I've finished *The Book* and was looking for what to do next.
Some infodump: I've never done anything programming related prior to ***Rust***, so assume my knowledge to be close to zero overall about everything (non-CS too, lol).

I've seen many people recommended web app as a project, and I wouldn't mind making a site for my personal stuff *(although it seems that a web **app** is something more complicated than just a site)*.

My problem is: I have no idea what to do. Every thing I look up about just brings even more stuff in the scope, and then that stuff does the same, and I understand nothing.

I've gone through ***axum*** docs and the only thing I understood from it is how to setup routes with `GET` and `POST`, yet I have zero clue about what to make into functions for them (and honestly, any more complicated stuff even in ***Rust*** itself just flies over my head right now), or what to do *at all*.

I'm also feeling lacking in overall concepts, maybe. For example:
I don't know what to do with this at all, so anything that would require me to make something myself would just throw me into paralysis *(same as current situation)*.

With that said, I'm feeling lost and having zero real progress right now.
What do I do? Where do I look?
The only thing that I have in mind is going through ***Exercism***, but everyone is saying that a project should be the next step, yet I just inexcusably can't make any advancement there.

Any advice *(even giving up)* is appreciated. If it's unclear what I'm even rambling about, ask to elaborate on something concrete, cause I myself not sure what I am asking for here.

P.S. I'm goint to sleep right now, so I'll respond tomorrow (can't make myself post that otherwise), sorry.

P.S.S. Optional, not exactly related to topic

My aquintance is saying that I should ask about that somewhere, yet I think I should be able to solve it myself. Do you think there is someone definetly in the right in this situation, or it just depends?

Question: Is Rust a language for beginners?

jjpe March 3, 2023, 4:30pm 2

Niro:

> even giving up

Definitely don't do this, Rome wasn't built in a day.

Niro:

> I've never done anything programming related prior to ***Rust***, so assume my knowledge to be close to zero overall about everything (non-CS too, lol).

This is where part of the difficulty comes from for you. You see, even for people that have experience, Rust has a bit of a learning curve, especially w.r.t. ownership, borrowing and the borrow checker, and the trait system. I think I recall reading in one of the previous annual Rust surveys that it takes most people who already have experience with C/C++ about 4 months to really internalize the consequences of those areas of Rust.

Given that you don't have that background yet, those things come on top of all the concepts that are also present in other programming languages that you have yet to learn.

So my point here is, please be realistic in what's ahead of you: quite a lot.
Whether that's terrifying or liberating depends on your character.
And coming back to what I said at the start, just learn bits and pieces at a time. Don't try to stuff everything into your head at once, that won't work[1].

[1]In fact that's the main reason most people remember atrociously little about what they were supposed to have learned during high school and even university.

8 Likes

steffahn March 3, 2023, 4:59pm 3

If you enjoyed reading the book, and perhaps re-producing the code in the example chapters, just consuming more learning material is definitely a valid way to continue; it's not like you *have* to start programming yourself right away. I personally would consider the book as a small starting point anyways, there's still lots to be learned. A few further reading and/or video resourced are e.g. listed in this thread. Looking at the "crust of Rust" video playlist e.g. why not learn more about how to program sorting algorithms in Rust or learn more about iterators? While I enjoy them myself, I cannot really judge the amount of prior knowledge required

for understanding these videos well enough not to come out of them entirely confused, though maybe even with a high perceived difficulty, it's approachable if you take it slowly; e.g. for the sorting algorithms one, you could program along, re-producing the code, and possibly taking breaks to look into library documentation for anything that's used that you want to learn more about for a better understanding. (And, if need be, you can consult other learning resources for better understanding the sorting algorithms in question in the first place.)

I think the suggestion of "make a web app" might often be directed at people who might have prior experience with web development (or just a *huge* motivation to learn a bunch of new stuff all at once at the same time as learning a programming language). I would consider myself quite knowledgeable about the Rust language, but if you asked me to write a web app, I wouldn't even know where to start, not to speak about lack of motivation to do such a thing for no good reason.

Regarding sites like exercism, those can be great fun and good for actually *using* the language in small self-contained doses and when you might not want to do (or even know what to do as) any larger "real" project. That being said, you'll of course need to find exercises with the right difficulty level for you. Maybe even *more* easy than necessary, if possible, in order to get started with something at all. If something feels too hard, skip it! There's nothing lost, you could always come back later! I've taken a look at rustlings before, which is a rust-specific collection of exercises, though I don't quite remember how easy those were or weren't from the quick look I took at it some time ago. Searching for Rust exercises, I also came across this page now which I had never seen before, but apparently a good amount of people liked it on GitHub.

5 Likes

H2CO3 March 3, 2023, 5:23pm 4

Niro:

> The only thing that I have in mind is going through ***Exercism***, but everyone is saying that a project should be the next step, yet I just inexcusably can't make any advancement there.

You aren't obliged to take any particular advice. If you are not yet ready to make your own project from scratch, then don't.

Exercism has fun little problems, which could help you get some positive feedback and encouragement by rewarding the progress you make step-by-step. That might be perfect if you are just getting started.

Along with solving hands-on problems, I'd *strongly* encourage you to read up on some theory, too. To get started with programming (regardless of the particular language chosen), a typical university curriculum would take you through the following, at the very minimum:

- the basics of algorithms, the difference between an algorithm, a program, and code written a programming language, the significance of stored-program Neumann computers
- rudimentary math needed for understanding digital logic (boolean operations, integer representations, arithmetic and bitwise operations); this will probably lead on to a tangent about the history of computer hardware architecture (data representations were historically chosen to be efficient and/or easy to work with by the spcific computers of their time); you'll likely bump into the fun that is floating-point, too
- basic structure and elements of imperative programs: expressions, variables, conditionals/branching, loops, functions, types, records
- Fundamental data structures and algorithms (searching, sorting, trees, hash tables, …)
- the use of 3rd-party code, libraries, imports
- elementary skills in the command line (I like to call this "IT culture" or more broadly as a sub-topic of "digital literacy") – how to invoke a command, navigate directories, compile and run your own code, use SSH, install software, search text files
- Later, the difference between major paradigms (imperative, OO, functional, declarative, …)
- At some point, an introduction to database management and SQL

The most important of these all is probably acquiring the basic skill of formalizing problems and devising (simple) algorithms to solve them. Unless you have a *feeling,* an instinctive piece of knowledge as to when you should use a loop, an if, or a function, it'll be extremely unlikely that you'll be able to write a complete web application from scratch.

A web application might also not be a good beginner's project, because it involves…:

- many (and I mean, *many*) layers of abstraction, a metric crapton of tooling, most of which is completely and utterly useless for learning the *essence* of programming or a language;
- a lot of technical issues (eg. browser compatibility, JavaScript engine versions and capabilities) that make your life unnecessarily harder unless you already navigate the concepts and tooling smoothly;
- certainly at least 4 or 5 languages (for you, Rust and some DB query language on the backend, but also inevitably HTML, CSS and JavaScript on the front-end, unless you want 12 pt black Times Roman on an irritating full white background) – if you aren't even proficient in one language and paradigm, then you'll find yourself completely clueless trying to learn 5 of them at the same time.

So, all in all:

- Take it slow, solve simple exercises if that's what you need at this stage; acquire a feeling for formulating algorithms to solve problems
- Learn the CS theory too, most importantly, the lower-level details and the use of the basic algorithms/data structures
- If a web app is too much for you, start with something less bloated/messy. Try writing a Pong clone, for example.

15 Likes

firebits.io March 3, 2023, 6:35pm 5

I would recommend you to define your goals and interests first: Why do you want to learn to code? And what do you want to code?

This is important because, as you know, programming is now a key knowledge in the modern digital world. The knowledge/resources you need to know how to create a videogame can differ greatly from those needed to develope a desktop or a web application.

Once you know your goals and what motivates you, then start defining the small tasks that will get you there.

Do you want to learn to code a videogame? Start with the basics of CS such as data structures and algorithms, and then try to create a clone of Pong! Do you want to learn how to create a web app? Start by learning html, then CSS, then JavaScript!

3 Likes

jbe March 3, 2023, 7:04pm 6

I have to admit that I needed to go through learning material (including the book and others) over and over and over again. Sometimes when reading, it makes a lot of sense, but you quickly forget about it, and then you suddenly are confused about basic things like the difference between a `struct` and a `trait`, for example.

So my advice would be to go *slow speed*. Take time to review simple exercises over and over, then move on; and if you're stuck, then move back. Rust takes time to learn, at least it did for me.

With concrete problems or when some things in the teaching material don't make sense, the forum here is very helpful.

I personally made good experiences with the Rustlings course, it was a good challenge and forced me to get a deeper understanding (or move backwards to re-read the book, until I was ready to solve the exercises).

Regarding giving up: I gave up three times learning Rust, but finally I succeeded. I'm happy I came back.

If you want to move on to making some "real" programs, you could try creating a simple command line app, perhaps using `rustyline` and/or `clap`. Ask the user some information and do some calculations. That might be much easier than some GUI or network programming, yet can help you to understand Rust better in practice.

3 Likes

How do I even begin to learn Rust?

Niro March 4, 2023, 7:13am 7

Thanks for your replies, I'll go over some stuff in them.

[details=""Some stuff""]

jjpe:

> So my point here is, please be realistic in what's ahead of you: quite a lot.

Yeah, it's even clear from my current position that theoretically this area is never-ending stream of information that only keeps on growing, can only respect those people who stick with it.

steffahn:

> it’s not like you *have* to start programming yourself right away

Isn't it more of a practical skill?
As for me, I have some problems with theory in a sense that it doesn't really stuck if I don't actively use it (*and copying someone else's code is only useful for me to get a reference to it directly on my PC instead of opening the source, which I don't really mind)*.
For example: I've read some Postgres docs chapters, yet already forgotten almost everything cause I hadn't used it anywhere, and it's only been about 2 weeks.

steffahn:

> why not learn more about how to program sorting algorithms in Rust or learn more about iterators ?

Wow, those videos are pretty huge, not sure I could even muster that much concentration, especially if I'll meet the mentioned above problems. But I'll try, thanks.

H2CO3:

> If you are not yet ready to make your own project from scratch, then don't.

If I'm to judge, I'll never be ready

H2CO3:

> Take it slow, solve simple exercises if that's what you need at this stage

That's the problem, I have zero feeling on what I *need*.

H2CO3:

> Learn the CS theory too, most importantly, the lower-level details and the use of the basic algorithms/data structures

Are there any specific recommendations (*if I'm right to assume that means some literature)*?

firebits.io:

> I would recommend you to define your goals and interests first: Why do you want to learn to code? And what do you want to code?

Hitting at the core, huh.

I started as a way to get a career change for the most part, but looking at the overall situation with it, my hopes are kinda dimmed (*not to mention that knowledge of Rust in my country is needed only with 5+ years of experience in other languages, and it doesn't look much different worldwide*).

There's also a possibility to help (*do all the work for*) my friend, who got me involved with Rust at the first place, with his game on *Bevy*.

So overall, my motivations, as well as direction, are lacking, I guess, but I don't have anything more to go on in general in life, so...

firebits.io:

> Start by learning html, then CSS, then JavaScript!

Thought that would be the case, thanks for confirming.
Although I though that there's some stuff with **WebAssembly** to replace that with some implementations in **Rust** *(like **Yew**)*. Guess I didn't delve deeper.

jbe:

> Sometimes when reading, it makes a lot of sense, but you quickly forget about it

So true.

jbe:

> If you want to move on to making some "real" programs, you could try creating a simple command line app

I don't quite understand what CL app is. Should it be executable to be considered such? Does running your code from CL counts (*probably not, but won't mind to be certain*)?
[/details]

To summarize, the main options are:

- Go deeper in basic theories
- Go through **Rustlings**
- **Exercism** is okay-ish
- Try to make **Pong** and/or some other CLI app
- Learn frontend pack (all roads lead to JS).

Well, that's plenty to go on, thanks again.

1 Like

steffahn March 4, 2023, 7:35am 8

Niro:

> As for me, I have some problems with theory in a sense that it doesn't really stuck if I don't actively use it (*and copying someone else's code is only useful for me to get a reference to it directly on my PC instead of opening the source, which I don't really mind)*.

Regarding "copying" code, what I meant is something I've seen a few people do that recorded their experience of reading the rust programming language book: they made sure to follow along with example code by *typing it out themselves completely*, i. e. no copy-paste. The idea is that typing it out yourself could help you get familiar with the exact syntax of the code at hand better than if you copy-paste, and the inevitable typos you are going to make also would have you familiarize somewhat with reading and understanding compiler error messages.

Niro:

> Wow, those videos are pretty huge, not sure I could even muster that much concentration, especially if I'll meet the mentioned above problems. But I'll try, thanks.

Yes they're quite lengthy. But also they are live stream recordings, so the information density is less than in educational scripted videos, and also many of them cover a lot of topics or aspects of a topic, so it's possible to watch them part by part on different days

2 Likes

Lambdax March 4, 2023, 10:41am 9

Also a newbro here. coming from a high level language, im at the start of chapter 13 of the book.

Im not so much a fan of reading a book, but more so of experimenting and getting things to work that i didnt think would be possible. Which is something i already do between the chapters, because otherwise i would become bored. And this way i understand things that didnt make any sense to me while reading. Its like riding a bicycle instead of just reading about it - the learning process is better atleast for me.

Anway i already have in mind what i will do once i finished the book, which is to check out Rustlings and then i want todo basic tasks.

1. Like create simple guis
2. checking out crates and work with them
3. build a useless tcp/udp api
4. build a tool that fetches data from a public api
5. build a tcp http/s proxy (cause i did that in a high level lang)
6. build a very basic website and see how cloudflare and all that stuff works (as i want todo things with public webservers in the future)
7. Learn and stay true to the convention

And while i do i use my new gained knowledge to optimize already present high level projects. And from there on i hope to be able to recreate my high level projects within rust. I would be very proud of myself if i can fully relly on rust in the future

Have a nice one o/

1 Like

anon80458984 March 4, 2023, 11:58am 10

Niro:

> With that said, I'm feeling lost and having zero real progress right now.
> What do I do? Where do I look?
> The only thing that I have in mind is going through ***Exercism***, but everyone is saying that a project should be the next step, yet I just inexcusably can't make any advancement there.

I'm a big big fan of just brute force typing out every example in Rust by Example. IIRC, fixing the typos I introduced / modifying the examples / getting compiler errors / fixing compiler errors is what ended up giving me the confidence that I was sufficiently skilled in the language.

That, and I don't see you mention what editor / IDE you are using -- for me, IntelliJ + Rust plugin was another huge change. Being able to hit goto-def is often much better than reading docs.rs

2 Likes

jbe March 4, 2023, 1:32pm 11

Niro:

> jbe:
>
> > If you want to move on to making some "real" programs, you could try creating a simple command line app
>
> I don't quite understand what CL app is. Should it be executable to be considered such? Does running your code from CL counts (*probably not, but won't mind to be certain*)?

My idea was that you could write a simple program that creates text output and reads some text input from the user. See this post for a very very simple example. Of course, you could go more complex quickly.

But I would say: Better writing simple programs that you master instead of copy&pasting complex code which you don't really understand. (At least that's my p.o.v., though I have to admit that other people can be incredibly productive by recombining bits and pieces which they don't fully understand.)

3 Likes

anon80458984 March 4, 2023, 3:18pm 12

jbe:

> But I would say: Better writing simple programs that you master instead of copy&pasting complex code which you don't really understand. (At least that's my p.o.v., though I have to admit that other people can be incredibly productive by recombining bits and pieces which they don't fully understand.)

This is probably better than what I wrote above. If you're already at the point where you're familiar with basic Rust concepts / fixing rustc errors, @jbe 's advice is probably better.

1 Like

Niro March 5, 2023, 5:18am 13

steffahn:

> *typing it out themselves completely*, i. e. no copy-paste

Yeah, I've been doing that too (most of the times)

anon80458984:

> I don't see you mention what editor / IDE you are usin

VSCode, with rust-analyzer as well.

jbe:

> See this post for a very very simple example.

So does Book's guessing game counts as CLI programm? Cause I was doing most of other homeworks with it as a base, so perhaps I've made something unknowingly already? That would be funny.
If so, guess I could upgrade it with clippy/rustyline

stonerfish March 6, 2023, 12:13am 14

Yes the game is a CLI.
Command line program means a program you run in a terminal and has standard input and output. *almost all programs, even gui's can be started as a command in a terminal*
You are doing fine, maybe just getting bored because the cli examples and web server/client interactions, string processing, regex, stuff is boring. Go sideways for a bit and play with something that is "fun"
I suggest trying to make something "pretty"

Start looking at some programs in the "examples" folder of crates that are "fun" and find a short code to play with.
For example rust_minifb, julia example is kind of short code. When run, it draws something pretty. Get it (or another example from some other crate that looks "fun" to you) to compile and run. Then modify the code to make the window bigger. Modify the code to do some other "math" , draw something else. Look at the fun graphic you create and relax until you think of something else to hack at.

2 Likes

jbe March 6, 2023, 12:32am 15

Niro:

> So does Book's guessing game counts as CLI programm?

Yes, like @stonerfish said, that's a command line program, so you have been doing something like that already.

Generally you can extend command line programs with a lot of different things. To give a few examples (of which some are easier and some are more complex):

- Interpret command line options (using `clap`)
- Read and write files from the file system (be careful with writing or deleting files though!)
- Do network requests (likely this is a bit more advanced already)
- Work as network server (e.g. a webserver)
- Open graphical windows
- Convert data, e.g. make the contents of a file all UPPERCASE CHARACTERS

(just some ideas of what you could try to do)

TechToFuji March 6, 2023, 7:09am 16

Hi @Niro you are in the right place

I recommend reading Zero to Production if you want to unstuck and get to the next level.
https://www.zero2prod.com

Niro March 6, 2023, 8:54am 17

stonerfish:

> Start looking at some programs in the "examples" folder of crates that are "fun" and find a short code to play with.

jbe:

> (just some ideas of what you could try to do)

I'll try this stuff out, thanks

TechToFuji:

> I recommend reading Zero to Production if you want to unstuck and get to the next level

Will check it out, thanks, looks like what I want in theory

But only what's available, don't see my financial situation improving anytime soon to afford it

mnowicky March 7, 2023, 3:32am 18

Like everyone else has said already, you're missing the fundamentals. Regardless of what language you're interested in learning, much of the differences are just syntax. Rust has some unique characteristics to it, so I wouldn't necessarily recommend it as a first language, but it's too late for all that. It would almost be better if coding books described fundamental topics and gave examples in various languages rather than being language specific and starting with the same dry beginnings (types eg. strings vs numerals vs floating point etc., operators, conditionals, whatever).

You're going to want to fully understand functions and classes (traits), OOP (even though structs and enums with methods aren't called "objects" in rust, it's the same functionality). The problem with rust as a first is that all of these things that exist in most every language, and called the same thing, are called something else in rust. If you can get your head around that, it'll do you some good.

Some languages that might be better suited for first timers are JavaScript/node, python, or C#. I came to rust looking for something low-level that I could compile directly into executables, which rust is great at (rather than something high level/ interpereted). This is a rust forum so I wouldn't recommend you go try those instead, but if you're on the verge of giving up on coding altogether, maybe try your hand at one of those.

Any general purpose language will do if you just want to understand programming concepts and progress. There's a game called "Screeps" that is fantastic- it's basically just an API made into a territory defense/resource management game and you have to code all of your own functionality. Maybe have a look as it's great fun and will make you a better coder by leaps and bounds if you enjoy it.

jbe March 7, 2023, 4:08am 19

I'm not sure if Rust is per-se a bad language to start learning programming. I think its strictness in regard to mutability, for example, may also help to get a deeper understanding for certain problems than if you'd use a language where everything is reference counted, garbage collected, type erased, etc.

But I'm not sure.

I feel like the existing material to learn Rust from a total beginner's perspective might be insufficient. That doesn't make the material bad because most people coming to Rust *will* have prior programming experiences. But it's not suitable as a "learn what programming is about" course.

I do believe, however, that it would be possible and maybe even beneficial to teach the art of programming with Rust.

2 Likes

ZiCog March 7, 2023, 9:39am 20

mnowicky:

> It would almost be better if coding books described fundamental topics and gave examples in various languages rather than being language specific

I think that for somebody totally new to programming being presented with examples in multiple languages would be very confusing.

mnowicky:

> Some languages that might be better suited for first timers are JavaScript/node, python, or C#.

There has been much debate about that here. Having started out back in the day with BASIC, soon followed by assembler than ALGOL I was kind of inclined to agree. Now I'm not so sure.

A raw beginner needs to grasp the basic concepts of what a program even is. That is basically "sequence", "selection" and "iteration". That is to say sequences of some kind of instructions/commands/statements, selection of alternative sequences with some kind of "if", repletion of such sequences with some kind of loop.

Of course before that makes any sense our beginner will need to grasp variables, operators and expressions.

The we can move our beginner up to higher level things like structures, functions, traits etc.

I believe all of this introductory stuff could be done just as well in Rust as JS, Python etc. After all a lot of Rust that I write looks pretty much like JS.

An important practical thing for introducing a beginner is having a programming system that is very easy to get started with and can get them from typing a little code to a running result with the minimum amount of hassle and conceptual overhead. Of course Python and others are great for that (As was BASIC back in the day). But I would argue that getting a minimal "hello world" program running with 'cargo run' fits the bill very well.

The only issue is that I have yet to see any Rust learning materials that are targeted at such raw beginners to programming. Likely that will come.

mnowicky:

> You're going to want to fully understand functions and classes (traits), OOP (even though structs and enums with methods aren't called "objects" in rust, it's the same functionality).

I have to disagree there. Rust is not an Object Oriented Programming language. It really does not support basic OOP things like classes and inheritance. Traits are not really like classes. A such I think it's appropriate that stunts, methods, etc don't get referred to as "objects", that would imply to much OOP of Rust which it does not have.

mnowicky:

> The problem with rust as a first is that all of these things that exist in most every language, and called the same thing, are called something else in rust.

I do not see this as a problem, for a raw beginner. They won't be confused by any inappropriate terminology they have leaden elsewhere (like OOP terminology).

Even for experience users of other languages it's good that Rust does call things by different names, it's a different language, the things are not so similar (see above).

4 Likes

next page →

Related topics

Topic		Replies	Views	Activity
I know nothing about programming or advanced computer stuff	30	748	July 21, 2025	
There are no ideas and imagination for developing projects	19	531	September 13, 2025	
help				
Complete Noob to coding Question	23	1474	October 19, 2021	
help				
Newcomer's impressions - improve learning experience	16	1062	March 17, 2024	
community				
I'm new to rust	18	657	September 23, 2025	
help				