main.rs

#

Welcome to Rust-101

This is Rust-101, a small tutorial for the Rust language. It is intended to be an interactive, hands-on course: I believe the only way to *really* learn a language is to write code in it, so you should be coding during the course.

If you have any questions that are not answered here, check out the "Additional Resources" below. In particular, the IRC channel is filled with awesome people willing to help you! I spent lots of time there ;-)

I will assume some familiarity with programming, and hence not explain the basic concepts common to most languages. Instead, I will focus on what makes Rust special.

Why Rust?

When you got here, I am kind of assuming that you already decided to give Rust at least a look, so that I don't have to do much convincing here. But just in case, here's why I think Rust is worth learning:
At this time, Rust is a language with a pretty unique set of goals. Rust aims to achieve C++-style control over memory and execution behavior (like, static vs. dynamic dispatch), which makes it possible to construct abstractions that carry no run-time cost. This is combined with the comfort of high-level functional languages and guaranteed safety (as in, the program will not crash in uncontrolled ways). The vast majority of existing languages sacrifices control for safety (for example, by enforcing the usage of a garbage collector) or vice versa. Rust can run without dynamic allocation (i.e., without a heap), and even without an operating system. In fact, Rust rules out more classes of bugs than languages that achieve safety with a garbage collector: Besides dangling pointers and double-free, Rust also prevents issues such as iterator invalidation and data races. Finally, it cleans up behind you, and deallocates resources (memory, but also file descriptors and really anything) when you don't need them anymore.

Getting started

You will need to have Rust installed, of course. It is available for download on the Rust website. Make sure you get at least version 1.3. More detailed installation instructions are provided in The Book. This will also install `cargo`, the tool responsible for building rust projects (or *crates*).

Next, we have to prepare a workspace for you to conduct your Rust-101 work in, so that you don't have to start with an empty file. The easiest way is to download the workspace matching the online tutorial. Try `cargo build` in that new folder to check that compiling your workspace succeeds. (You can also execute it with `cargo run`, but you'll need to do some work before this does anything useful.)

Alternatively, you can build the workspace from source by fetching the git repository and running `make workspace`.

```rust
#![allow(dead_code, unused_imports, unused_variables, unused_mut, unreachable_code)]
mod part00;
mod part01;
mod part02;
mod part03;
mod part04;
mod part05;
mod part06;
mod part07;
mod part08;
mod part09;
mod part10;
mod part11;
mod part12;
mod part13;
mod part14;
mod part15;
mod part16;
```

Course Content

Open `your-workspace/src/part00.rs` in your favorite editor, and follow the link below for the explanations and exercises. You are ready to start. Have fun!

Introduction

- Part 00: Algebraic datatypes
- Part 01: Expressions, Inherent methods
- Part 02: Generic types, Traits
- Part 03: Input

Basic Rust

- Part 04: Ownership, Borrowing, References
- Part 05: Clone
- Part 06: Copy, Lifetimes
- Part 07: Operator Overloading, Tests, Formating
- Part 08: Associated Types, Modules
- Part 09: Iterators
- Part 10: Closures

Advanced Rust

- Part 11: Trait Objects, Box, Lifetime bounds
- Part 12: Rc, Interior Mutability, Cell, RefCell
- Part 13: Concurrency, Arc, Send
- Part 14: Slices, Arrays, External Dependencies
- Part 15: Mutex, Interior Mutability (cont.), RwLock, Sync
- Part 16: Unsafe Rust, Drop

To actually run the code of some part (after filling in the blanks, if necessary), simply edit the `main` function.

```rust
fn main() {
    part00::main();
}
```

Additional material

There's tons of useful Rust stuff out there, so let me just put links to some of the most interesting places here:

- The Rust Book
- The Rustonomicon
- Rust by Example
- The Rust Subreddit
- A collection of links to blog posts, articles, videos, etc. for learning Rust.
- For the IRC channel and other forums, see the "Community" section of the Rust Documentation index