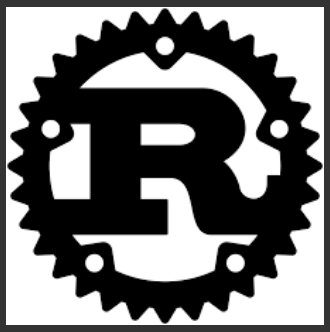
VOCATIONAL RUST 2/2

This two-module-long Rust course is made for developers that already have some experience in functional programming and wish to build upon it an advanced expertise in Rust. As a low-level language, it ties robustness and performance in one package which makes it a serious contender to replace C and C++.
This second module dives into advanced aspects of traits and ownership, which paves the way to discussing more advanced notions such as boxing, concurrency, collections... All sections include exercises that participants solve in-session, with help from the trainer(s). Also, this module builds on a running example so that participants follow and contribute to a growing codebase performing non-trivial tasks. The final version is about 3.5 KLOC.



INFORMATION

Price: 2000 EUR excl. taxes
Duration: 3 days
Practice: 50%

Public
Developers

Pre-requisites
Functional Programming
Rust module 1

Pedagogical Objectives

- Customization of the program

OPT Declarative macros

- Advanced traits
- Advanced ownership
- More basic (niche) notions
- Type size and boxing
- Collections
- Interior Mutability

OPT Single Dispatch

- Fearless Concurrency

OPT Unsafe Rust and FFI

Langages
French
English

Indicateurs de résultats
Pas encore disponibles.

CONTACT

contact@ocamlpro.com
+33 6 72 73 37 53
21, rue de Chatillon, 75014, Paris, France
training.ocamlpro.com
@ocamlpro

Version: November 15, 2022

TRAINERS

Adrien Champion (Rust)

Adrien is a senior R&D developer at OCamlPro since 2018 after a PhD in Computer Science and a post-doc in Japan. Adrien develops in Rust since the first versions of the Rust compiler, and maintains several open-source crates in Rust, such as hashconsing, rsmt2, zdd ou safe_index.

TRAINING PROGRAM

Customization of the program

Some sections are labeled as [OPT]ional. They cover topics that participants will be fine learning about on their own using what they learnt during the course. They can do so either from the course's material (which include exercices for all optional sections) and/or from alternative resources online. These sections are optional because there is simply not enough time to cover everything in three days. Different clients will want to prioritize different concepts, which is why we propose to tailor the course; it can either contain

- all optional sections: the trainer will present them without the exercises;
- one optional section: fully presented, with exercises;
- a tailored version of the above, to discuss before training starts.

Alternatively, clients can elect not to include any optional section. While this can sound like a net loss, trainers adapt the speed at which they introduce notions and conduct exercises in real time. Not having any optional section gives more time to discuss Rust's fundamental and advanced concepts, and practice them. As mentioned already, none of the optional sections are very difficult to dive in for anyone with a good grasp on Rust's main concepts. As a consequence it might be more efficient to take the time to build solid foundations.

[OPT] Declarative macros

- Writing a declarative macro
- Visibility
- Hygiene

Advanced traits

- Trait coherence rule
- Fundamental traits from the standard library
- Complex trait bounds

(continued on page 2)

VOCATIONAL RUST 2/2 (PAGE 2)

Advanced ownership

- Dive into borrow rules
- Lifetime subtyping
- Storing references
- Implementing traits for references

More basic (niche) notions

- Global state: const and static
- Turbo fish

Type size and boxing

- Sized types and DSTs
- Algebraic datatypes
- Reference counting

Collections

- Fundamental types: Vec, HashMap, BTreeMap
- Fundamental traits: Iterator, IntoIterator, Collect

Interior Mutability

- Detailed discussion of RefCell
- RefCell limitations (using rayon)

[OPT] Single Dispatch

- Trait objects
- Heterogeneous collections
- Lifetime bounds for trait objects

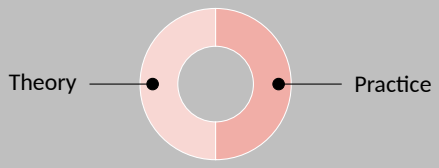
Fearless Concurrency

- Rc and Arc
- Send and Sync
- Spawning threads and passing messages

[OPT] Unsafe Rust and FFI

- Unsafe powers
- Raw pointers
- Unsafe functions and traits
- FFI basics
- Calling Rust from C and C from Rust

THEORY VS PRACTICE



USEFUL INFORMATION

EVALUATING PROGRESS

We make the progress of our trainees and its evaluation a core aspect of our courses. Indeed, guaranteeing the durable acquisition of the skills at hand is key, especially for the newer comers. To that extent, we will have trainees undergo tailored group works, exercises and hands-on practice which modalities can all be customised to your specific needs.
At the end of the course, you will have an opportunity for feedback to help us improve upon our methods. This is crucial as we believe there is always room for learning on both sides of the desk and no opinion other than yours matters more for us to do so.

CONSIDERING RQTH(RECOGNITION OF HANDICAPPED WORKER STATUS)

If people with disabilities are part of the course, do reach out to us so we can adapt the training accordingly.

PEDAGOGICAL RESSOURCES

The ressources are written by the OCamlPro team prior to the courses. Documents are generally written in english and can be translated to french if need be.

FUNDING RESORTS IN FRANCE

Unfortunately our trainings cannot yet be funded by institutions such as OCPO (despite these funds fully covering the price of our trainings), neither can they be funded by CPF.

INTER CORPORATION TRAININGS

Les horaires pour nos formations inter-entreprises in-situ sont :
Start - 9:30AM Lunch Break - 12:00PM to 01:00PM End - 05:30PM