

Rust Design Patterns

Audience Course Rust Design Patterns

The course Rust Design Patterns is intended for Rust developers and software architects who want to apply Rust Idioms and Design Patterns when designing Rust applications.

Prerequisites Course Rust Design Patterns

Knowledge of and experience with Rust is required. Experience with object oriented analysis and design with UML is recommended.

Realization Training Rust Design Patterns

The concepts are covered on the basis of presentation slides. The theory is illustrated with demos of patterns in Rust. There are exercises in design problems where Rust patterns are applied.

Certificate Course Rust Design Patterns

After successful completion of the course the participants receive a certificate Rust Design Patterns.

Duration: 3 days

Price: € 2250

[Open Schedule](#)



Content Course Rust Design Patterns

In the course Rust Design Patterns the participants learn about design patterns and idioms that specific for the Rust language. Design patterns form a general repeatable solution to a commonly occurring problem in software design. Design patterns can be considered templates for how to solve a particular problem.

Rust Recap

The course Rust Design Patterns starts by reviewing the important features of the Rust language, including Ownership, Moves, Shadowing, Guards, Crates, Closures and Traits.

Rust Idioms

Next attention is paid to the idioms and conventions that are specific to Rust programming. This will include idioms like Borrowed Type Arguments, Collections as Smart Pointers, Finalization in Destructors and On-Stack Dynamic Dispatch.

Behavioral Patterns

Subsequently Behavioral Patterns that are related to object behavior and communication, are explored. The specific Rust implementation of familiar patterns like the Observer pattern, the Command pattern and the Iterator pattern, are discussed.

Structural Patterns

The course Rust Design Patterns also pays attention to structural design patterns that are concerned with the structure of objects and classes. This includes the Rust specific implementation of patterns like the Adapter pattern, the Façade pattern and the Composite pattern.

Functional Programming

Finally functional programming patterns and how they can be applied in Rust, are on the schedule of the course Rust Design Patterns. This will include topics such as Object Based API's, Type Consolidation and Wrapping Iterators.

Modules Course Rust Design Patterns

| Module 1 : Rust Recap | Module 2 : Rust Idioms | Module 3 : Behavioral Patterns |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Rust Data Types Ownership and Moves Type Anonymity Shadowing Guards Crates Closures Traits Designators Lifetimes Dynamic Dispatch | Borrowed Type Arguments Strings with format! Constructor Default Trait Collections as Smart Pointers Finalization in Destructors On-Stack Dynamic Dispatch Iterating over Option Pass Variables to Closure Privacy For Extensibility Foreign Function Interface | Command Interpreter Newtype RAII Guards Strategy Visitor Chain of Responsibility Mediator Observer Iterator Strategy |
| Module 4 : Structural Patterns | Module 5 : Functional Programming | |
| Adapter Composite Decorator Bridge Façade Builder Factory Method Compose Structs Prefer Small Crates Small modules | Object Based API's Type Consolidation Wrapping Iterators Programming Paradigms Generics as Type Classes Lenses and Prisms Profunctor Optics Anti Pattern Unneeded Clone Defer Polymorphism | |